# An Experimental Platform for Heterogeneous Multi-Vehicle Missions

## Rapidly enabling complex, fault-tolerant behaviours in a real-world environment

*Eric Schoof, Chris Manzie, Iman Shames, Airlie Chapman, Denny Oetomo*
University of Melbourne
Melbourne, Australia
{eschoof, manziec, iman.shames, airlie.chapman, doetomo} @unimelb.edu.au

***Abstract*—This paper documents the development of an experimental platform that can be used to test and validate collaborative multi-vehicle algorithms and to explore the collective behaviors that arise. Unlike existing platforms, sensing and communication are explicitly considered, enabling realistic scenarios to be tested and prototyping of systems to be facilitated, thereby accelerating the development of next generation systems for defense and civilian applications.[1]**

***Keywords—coordinated autonomy; networked dynamic systems; teaming; multi-vehicle robotics***

## I. INTRODUCTION

Research into autonomous robotic systems has been ongoing for many years. Coordinated autonomous systems that allow many agents to cooperatively execute tasks are still in their infancy but are highly desired in many settings. For example, using a single vehicle in a search and rescue application means the mission area must be explored serially, and if the vehicle fails, the mission will terminate prematurely. If instead multiple vehicles can be used as in [1]–[3], then the mission area can be explored more effectively whilst the loss of any one vehicle can be compensated by the remaining vehicles to retain mission completion goals. Other examples include the missile seduction problem using multiple Unmanned Aerial Vehicles (UAVs) [4].

The simplest form of communication amongst multiple agents is to assume all agents can directly communicate with one another and all decision making is performed at a single location. This is undesirable as the tractability of the approach decreases with the number of agents and furthermore it introduces a single point of failure for the mission, namely the centralized computing node. The alternative (and more intuitive) approach is each agent is a member of a coordinated autonomous system in which communication and computation requirements depend only on the agents around them. Information spreads through the network over ever-changing active communication links in a decentralised manner, thus requiring that collective behaviour amongst the multiple vehicles satisfies the overall mission objectives.

To ensure mission objectives are met, the algorithms used within individual agents in an autonomous robotic system, and their collective behaviour, will need provable guarantees of their performance in this complex environment. This collective behaviour should then be validated in an appropriate emulation facility. To achieve this goal requires the analysis of networked dynamic systems [5]–[7], multi-robot simultaneous localization and mapping (SLAM) [8]–[12], fault-tolerant communication networks [13], [14], and clock synchronization [15]–[17]. Whilst individually these tasks have received different levels of attention, to date there is limited (if any) examples of combining these capabilities into a single platform for prototyping collaborative emergent behaviour of multiple robotic agents.

When developing new algorithms targeting multi-vehicle environments, it can be difficult to capture real-world behaviour as there is often a disconnect between scientific simulations, and real-world execution. This is mainly due to the inherent difficulty of operating many vehicles with potentially faulty algorithms, and the similarly inherent low-fidelity models used to simulate such groups of vehicles. Many groups have developed distributed hardware platforms to look at various research questions. For example, the Kilobot platform by K-Team was one of the first platforms to scale to large numbers of autonomous agents [18]. Each agent costs only a few dollars, has limited sensing and communicates with agents in their very-near proximity, and has a novel locomotion method involving micro vibrators to rotate and translate through the environment. The Kilobots, however, do not move or sense in a way that mirrors typical real-world scenarios, and the on-board computation limits what an individual agent can do on its own. The Robotarium at Georgia Tech [19], [20] is probably one of the most novel distributed hardware-in-the-loop (HIL) simulators, as it allows submission of code to be run on a real group of ground and aerial robots and will produce a recording of their behaviour. However, all computation and sensing is performed centrally, making it difficult to extrapolate the performance of the experiment to true distributed settings.

The University of Melbourne is developing a unique software and hardware platform to design and test multi-vehicle coordination algorithms in an environment that is both simple to use while still rich enough to capture many attributes of the real world. There are multiple levels of algorithm execution, including pure software, mixed HIL, and pure

---

[1] The authors acknowledge the support of the Competitive Evaluation Research Agreement (CERA) grant, “Shared Control Framework for Collaborative Human Robot Teams”.

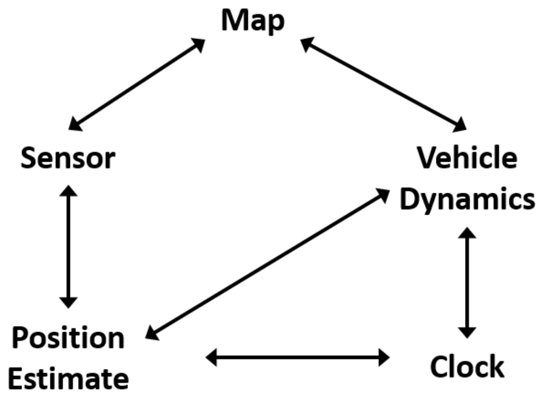

**Figure 1. The platform is split into five logical components, each of which can independently be run in either simulation or with hardware-in-the-loop.**

hardware – all of which combine to produce an effective platform for the testing and validation of collaborative multi-vehicle teaming algorithms.

This paper outlines the development of the University of Melbourne distributed robotics platform to date, as well as the developments we intend to undertake in the future. The final platform will enable rapid prototyping and validation of collaborative algorithms for cooperative missions in realistic settings and dovetails the algorithmic developments from the University of Melbourne, other leading universities, and industrial partners.

The structure of the paper is as follows. In §II, we discuss the architectural overview of our platform. The existing hardware and software developments are presented in §III with an accompanying proof-of-concept experiment with sample results in §III.C. Finally, concluding remarks and future directions are discussed in §IV.

## II. ARCHITECTURAL OVERVIEW

In a high-level context, there are several subsystems that we consider, which can be seen in Figure 1. Each of these subsystems can be run either in simulation or with hardware-in-the-loop, to aid in rapid prototyping.

The most readily simulated subsystems are vehicle dynamics and position estimation. There are many packages available, which incorporate high-fidelity models for many commonly used vehicles and sensors, and general physics simulators, such as Gazebo [21] and nVidia Isaac. These systems aim to replicate the world in a very high fidelity, both in terms of physics and sensing. They also typically require large amounts of computational power, and setup can be complicated.

The University of Melbourne robotic platform develops in parallel the hardware and accompanying software simulation of the testbed. We explicitly do not use the simulation packages above because our focus is on the high-level algorithmic development, and we want to keep all software bundles as a single package. This provides a powerful prototyping and testing environment for distributed algorithms.

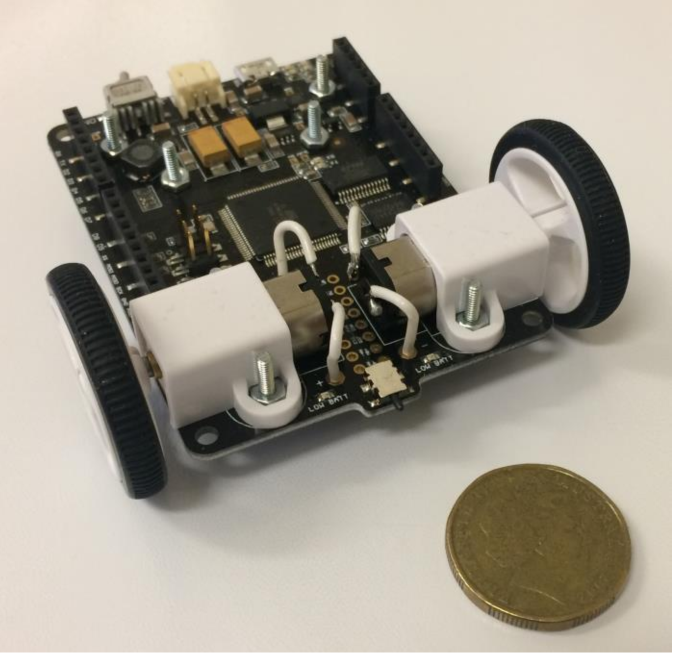

**Figure 2. Low-cost ground vehicle which can represent many real-world vehicles. Shown next to a $A1 coin for scale.**

## III. EXISTING DEVELOPMENTS

Currently we have developed several hardware modules and a software infrastructure to smoothly transition from pure simulation to HIL environments. In this section, we explore both, and give some insights into what we will do in the future to paint a broader picture of the final vision.

### *A. Hardware*

The hardware developed so far consists of a unicycle-style ground vehicle (Figure 2) and two custom interchangeable sensor modules: a low-cost laser ranging sensor (Figure 3) and an identification and relative localization beacon (Figure 4). We have opted to build our own hardware, rather than use off the shelf options, as we can strictly control the cost, computation, sensing, and communications requirements.

The main factor driving the design of the ground vehicle is to make it low cost; it is approximately $A80 for a single unit, and nearly half that in bulk. It is also functional in that it contains sensors and additional hardware to enable the unit to navigate its environment. Furthermore, the batteries are rechargeable and simple computation can be performed on-board with an 8-bit, 16MHz processor.

The laser rangefinder sensor module, seen in Figure 3, has eight discrete modules capable of centimetre accuracy ranging spanning 5cm - 180cm at a rate of 4Hz, sufficient for many mapping experiments. It has a much more capable 32-bit, 120MHz processor, and can do moderate processing on-board. Further, it has an XBee wireless radio to facilitate communication with other agents. It costs $A50 per unit.

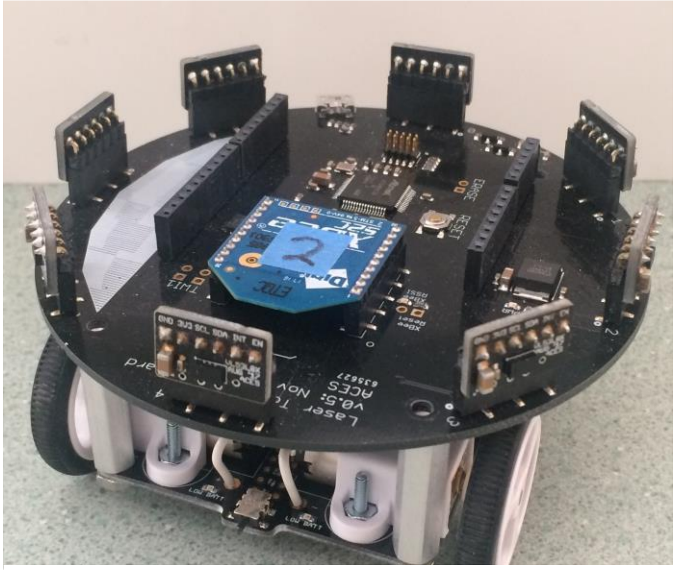

**Figure 3. Laser rangefinder module with eight solid-state sensors.**

Finally, the IR beacon, shown in Figure 4, can be used to sense neighbouring agents for the purposes of collision avoidance. It functions by emitting a modulated data stream from a series of IR LEDs. These streams can be detected using six detectors spaced around the sensor. The received signal strength can then be used to estimate the relative position and velocity of the sending agent.

This module also addresses the problem of assigning agent labels or identifiers within multi-vehicle systems. This issue occurs when many agents use a wireless network to communicate, leading to challenges in determining which neighbour is nearby and thereby complicating communication between specific pairs of agents. This sensor sidesteps that issue because an label, such as a network address, may be embedded in the data stream of the sensor. Higher-bandwidth communication devices can be used once the identifiers are obtained. This sensor also has a 32-bit 120MHz processor and an XBee radio, and costs $A50 for a single unit.

It is worth noting that the use of XBee radios in a laboratory environment can result in a network where all agents are connected to each other, which makes testing true multi-agent algorithms difficult. In our scenario, a variable attenuator can be added between the XBee IC and the antenna, significantly reducing the communication range, and allowing true distributed algorithms to be validated.

### *B. Software*

As our platform is intended to be used to rapidly develop and validate distributed algorithms, we chose to use MATLAB for its ease of use and prevalence in the control development community. Algorithms requiring an additional performance boost can typically be ported to C/C++ with minimal effort once the algorithm design has been confirmed.

One method to enable rapid prototyping of algorithms is to have both a simulated mode and a hardware-in-the-loop mode for each component. For example, a sensor will typically observe characteristics of the environment around it. If a map of the environment is provided *a priori* the sensor measurements may be approximated.

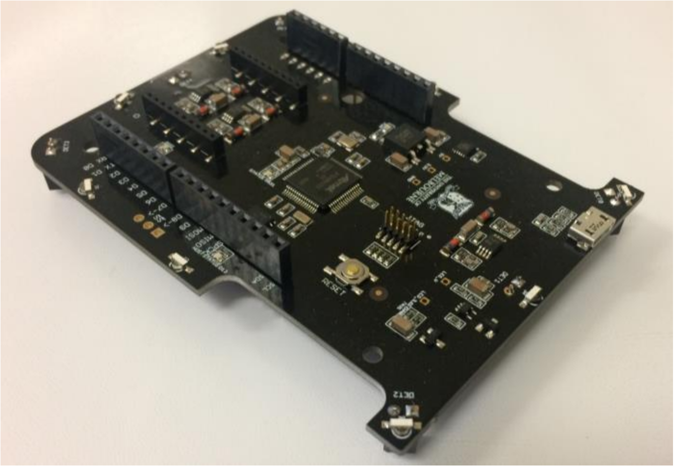

**Figure 4. IR Beacon used for identification and sensing relative agent positions.**

We have developed many MATLAB classes that can be used to structure and execute simulations. For example, codes have been written to communicate with both physical and simulated agents, allow localization by external motion capture, SLAM, and *a priori* recorded trajectories. There is also code to incorporate sensor observations from laser rangefinders, estimated sensor observations using a sensor model and a provided map, or replaying a recorded set of sensor observations.

The vehicles themselves can also be simulated in an analogous way using a model of their internal dynamics or using the physical vehicles in the loop. They may also be automatically controlled using trajectories provided *a priori*, or via a controller connected to a computer.

One often overlooked component is the scheduler or clock. Each device in the simulation environment has its own clock, which may operate at a different rate than the other clocks. This is because real-world clocks are imprecise and suffer from drift. While typically not noticeable over short time scales, this error can become problematic when integrated over longer periods. Further, most analytic guarantees on control algorithms are generally only satisfied when all agents observe each other at the same rate. For example, if one vehicle executes its observation and control loops much faster than its neighbours, it will be influenced more by the network than will the network by it. Consensus-style algorithms are particularly susceptible to clock drift, and many theoretical guarantees are only provided when agents are synchronized.

### *C. Initial Results*

We have currently tested our platform on some canonical mapping and localization examples. In Figure 5, we can see a mapping example where the range sensor in Figure 3 is simulated. In parallel, the real data from a physical sensor is used to construct a complimentary map on-board the ground vehicle. This example highlights the ability to toggle between

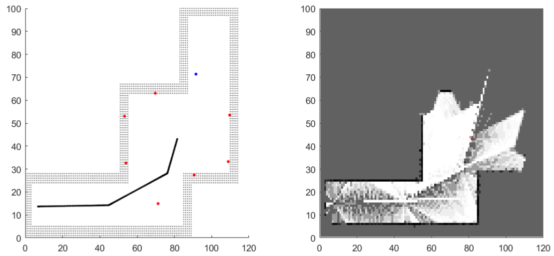

**Figure 5. Mapping example using simulated version of the laser rangefinder sensor.**

simulated and physical execution of the algorithm in a unified environment.

We are using this platform as a prototype for defence applications, such as enabling teams of robotic agents navigate uncertain environments like disaster areas to find wounded survivors.

## IV. CONCLUDING REMARKS AND FUTURE WORK

We have developed a preliminary platform and corresponding software stack for designing and testing highly-scalable multi-agent algorithms that perform well in non-laboratory settings. This was achieved by writing simple software classes that can run all aspects of an experiment in simulation, and that allow each component to also run independently in hardware. The software is installed as a single set of MATLAB files, allowing installation without significant customization, for example with ROS or a standalone simulator. The hardware itself is inexpensive, allowing experiments to cost-effectively scale to many agents.

There are several improvements that are planned for our platform. First, we will integrate coarse human inputs into the planning and control of the distributed autonomous robotic systems. Second, allowing humans and robots to interact with each other in predictable and intuitive ways will allow many real-world scenarios to be executed more naturally. We also intend to extend the hardware platform into the UAV space, both quadrotors and fixed-wing vehicles, and the unmanned underwater vehicle (UUV) space. Finally, we will create a more comprehensive dataset of maps and sensor models to facilitate testing a wider variety of configurations in simulation.

## REFERENCES

[1] Z. Yan, N. Jouandeau, and A. A. Cherif, “A survey and analysis of multi-robot coordination,” *Int. J. Adv. Robot. Syst.*, vol. 10, 2013.

[2] L. E. Parker, “Current State of the Art in Distributed Autonomous Mobile Robotics,” in *Distributed Autonomous Robotics Systems 4*, no. November 2002, G. Bekey and J. Barhen, Eds. Tokyo: Springer, 2000, pp. 3–12.

[3] R. M. Murray, “Recent Research in Cooperative Control of Multi-Vehicle Systems,” *J. Dyn. Syst. Meas. Control*, vol. 129, no. 5, pp. 571–583, 2007.

[4] I. Shames, A. Dostovalova, J. Kim, and H. Hmam, “Task Allocation and Motion Control for Thread-Seduction Decoys,” in *IEEE Conference on Decision and Control*, 2017.

[5] A. Chapman, E. Schoof, and M. Mesbahi, “Semi-autonomous networks: Theory and decentralized protocols,” *IEEE Int. Conf. Robot. Autom.*, pp. 1958–1963, May 2010.

[6] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton, New Jersey: Princeton University Press, 2010.

[7] E. Schoof, A. Chapman, and M. Mesbahi, “Bearing-compass formation control: A human-swarm interaction perspective,” in *American Control Conference*, 2014, pp. 3881–3886.

[8] M. Walter and J. Leonard, “An Experimental Investigation of Cooperative SLAM,” in *IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, 2004, vol. 37, pp. 880–885.

[9] B. D. Gouveia, D. Portugal, D. C. Silva, and L. Marques, “Computation Sharing in Distributed Robotic Systems: A Case Study on SLAM,” *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 410–422, 2015.

[10] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu, “Decentralized Cooperative SLAM for Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach,” *J. Intell. Robot. Syst.*, vol. 66, no. 3, pp. 321–342, 2012.

[11] S. Thrun and Y. Liu, “Multi-Robot SLAM with Sparse Extended Information Filers,” in *Robotics Research. The Eleventh Internation Symposium*, P. Dario and R. Chatila, Eds. Berlin, Heidelberg: Springer, 2005, pp. 254–266.

[12] S. J. Julier and J. K. Uhlmann, “Simultaneous localisation and map building using split covariance intersection,” in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems.*, 2001, vol. 3, pp. 1257–1262.

[13] H. Liu, H. Darabi, P. Banerjee, and J. Liu, “Survey of Wireless Indoor Positioning Techniques and Systems,” *IEEE Trans. Syst. Man Cybern.*, vol. 37, no. 6, pp. 1067–1080, 2007.

[14] D.-L. Yang, F. Liu, and Y.-D. Liang, “A Survey of the Internet of Things,” in *Computer Networks*, no. May, Elsevier B.V., 2010.

[15] A. Giridhar and P. R. Kumar, “Distributed Clock Synchronization over Wireless Networks: Algorithms and Analysis,” in *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006, pp. 4915–4920.

[16] S. Graham and P. R. Kumar, “Time in general-purpose control systems: The Control Time Protocol and an experimental evaluation,” in *Proceedings of the IEEE Conference on Decision and Control*, 2004, vol. 4, pp. 4004–4009.

[17] J. Elson, L. Girod, and D. Estrin, “Fine-grained network time synchronization using reference broadcasts,” *ACM SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, 2002.

[18] M. Rubenstein, C. Ahler, and R. Nagpal, “Kilobot: A low cost scalable robot system for collective behaviors,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 3293–3298, 2012.

[19] D. Pickem *et al.*, “The Robotarium: A Remotely Accessible Swarm Robotics Research Testbed,” in *IEEE International Conference on Robotics and Automation*, 2017.

[20] D. Pickem, M. Lee, and M. Egerstedt, “The GRITSBot in its Natural Habitat - A Multi-Robot Testbed,” in *IEEE Internation Conference on Robotics and Automation*, 2015, pp. 4062–4067.

[21] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004, vol. 3, pp. 2149–2154.